Attribute and Context-Aware Multi-Behavior Model for Unique-Item Recommendation

Shereen Elsayed[1], Ngoc Son Le[1], Ahmed Rashed[2], and Lars Schmidt-Thieme[1]

[1] Information Systems and Machine Learning Lab (ISMLL) & VWFS Data Analytics Research Center (VWFS DARC), University of Hildesheim, Hildesheim, Germany
{elsayed,sle,schmidt-thieme}@ismll.uni-hildesheim.de
[2] Volkswagen Financial Services AG, Braunschweig, Germany
{ahmed.galal.ahmed.rashed}@vwfs.com

Abstract. In the context of sequential recommendation, incorporating auxiliary information has consistently shown improvements in several scenarios. Some models focus on integrating item and user features, other approaches include context information. A notable area of growth is the multi-behavior recommendation, which considers the different user's behaviors to indicate their preferences. Current methods rely on graph-based models while other sequential multi-behavior models use transformers. However, none of these models take items or user features into consideration which causes a limited understanding of user preferences through different actions and item characteristics. In this paper, we propose an **A**ttribute and **C**ontext-aware **M**ulti-**B**ehavior model (**ACMB**) for unique item recommendation. This not only accounts for users' varying behaviors but also integrates the relevant attributes of items to enhance the understanding of user preferences. ACMB encodes the items with the respective attributes then it applies a hierarchical attention over the different behaviors separately, followed by attention across the entire input sequence to generate a comprehensive deep sequence representation. Extensive experiments on real-world Volkswagen Financial Services (VWFS) dataset demonstrate the significance of our proposed model over the current state-of-the-art attribute-aware sequential recommendation methods.

1 Introduction

In Volkswagen Financial Services (VWFS) Business-to-Bussiness (B2B) settings, vehicles are sold to dealers through auctions. Auction systems possess unique characteristics as items are sold in a fixed-price round followed by several bidding rounds. These items are considered unique because each can be sold only once within the auction system. This scenario can be framed as a sequential recommendation problem. However, it presents a significant challenge that items (vehicles) are not identified by fixed IDs but rather by their attributes (features). This makes the problem highly relevant to attribute-aware sequential models.

Modern platforms increasingly leverage not only transactional data but also auxiliary information to capture user preferences, enabling the creation of more

accurate recommendation systems. These systems, often referred to as multi-relational or multi-behavior recommendation models, integrate additional behavioral data to improve performance. For instance, in retail platforms, incorporating behaviors such as "add to cart" or "clicks" enhances model accuracy. Similarly, auction systems provide rich auxiliary information beyond sales data, such as bid histories and user click interactions. Incorporating these additional behaviors is crucial for accurately capturing user preferences in such complex environments. However, existing state-of-the-art multi-behavior recommendation methods [21, 22, 2] rely heavily on fixed item IDs, which makes them unsuitable for unique item recommendation scenarios like auction systems. Addressing this limitation requires adapting models to work effectively in attribute-based settings without relying on item IDs.

To address these challenges, we propose an attribute- and context-aware multi-behavior framework for unique item recommendation. Unlike earlier approaches to multi-behavior recommendation [7], [19], which depend on item IDs for item representation, our method eliminates this dependency. Instead, it leverages rich item attributes to construct item representations. Additionally, our framework integrates interaction context, such as time and user behavior type to provide a comprehensive understanding of user preferences. This combination enables our model to effectively tackle the challenges posed by unique item recommendation in auction systems. Our contributions can be summarized as follows:

- We propose the first attribute- and context-aware multi-behavior model, which employs hierarchical attention to capture comprehensive behavioral representations.
- We introduce a novel approach that leverages item attributes and auxiliary behaviors to effectively learn user preferences in auction systems.
- Extensive experiments on a VWFS real-world dataset demonstrate that our proposed model, ACMB, significantly outperforms state-of-the-art attribute-aware sequential recommendation models.

2 Related work

In our work, we focus mainly on the three branches of research, consisting of attribute and context-aware recommendation, sequential recommendation, and multi-behavior recommendation.

Attribute and Context-Aware Models is a substantial sector in recommendation systems that seek to improve the quality of recommendations by leveraging both item features and contextual information, moving beyond mere item IDs. Early approaches utilizing contextual features are the factorization machine-based methods such as FM [13]. Later the neural factorization machine (NFM) utilized deep neural networks for learning non-linear feature interactions [5]. Furthermore, DeepFM [4] is a popular context-aware method that combines the FM and DNNs for extracting latent representations. More recent approaches benefit from incorporating the time-aspect, which allows learning the sequential

pattern in the data along with the contextual features. More recent state-of-the-art attribute and context-aware sequential methods is CARCA model [11], which includes contextual information and item attributes. The sequence of profile items and the target items passes through a cross-attention mechanism to get the final items' scores. Additionally, ProxyRCA [14] further improves upon CARCA, by improving the items encoding method.

Sequential Recommendation is a prevalent task in recommender systems that utilizes the historical interactions of each user to predict the next item the user will most likely interact with. GRU4Rec [6] is one of the early approaches that utilize RNNs for mining the sequential behavior in a session-based recommendation scenario. Another sequential model is Caser [16], which applies convolutional filters on the time and item embeddings to capture the latent sequential behavior of users. Additionally, more recent approaches started to employ transformer architectures such as the SASRec [8] model, which feeds the item embedding sequence into a multi-head self-attention block to capture the sequential correlation between historical items. Another method that improves upon SASRec is BERT4Rec [15], which uses a Bi-directional self-attention mechanism to better model the sequential behaviors. Other recent approaches tried to extend the SASRec model are SSE-PT [23], TiSASRec [10] and S^3Rec [17].

With the current massive data sources available for recommender systems, recent models started not only to employ the user-item primary interaction relation such as purchases but also to use other available click data. Employing auxiliary information has different namings in the existing prior work: relation-aware recommender systems, multi-relation recommendation, and multi-behavior recommendation.

Many recent **multi-behavior recommendation** approaches rely on convolutional graph networks by treating the different behaviors as a global heterogeneous graph, such as the MB-GCN [7] and MB-GMN [20] models. Additionally, other methods tried to employ transformer encoders in their architecture for better representation learning. One such example is the memory-augmented transformer networks (MATN) [18], which uses a transformer encoder followed by a cross-behavior aggregation layer to model the different behaviors. Moreover, KHGT [19] is a graph transformer method that captures user-item interaction characteristics; also, a graph attention layer is employed to understand the item-item relation further. State-of-the-art advancements include MBHT [21], which utilizes low-rank self-attention coupled with a hyper-graph neural architecture to model long- and short-term dependencies. MB-STR [22] introduces a multi-behavior sequential transformer layer to simultaneously learn sequential patterns across various historical behaviors. Furthermore, MBSRec [3] learns all behaviors via a multi-head self-attention block while maintaining the behavior contibution using weighted binary cross-entropy loss.A dditionally, HMAR [2] employs a hierarchical attention mechanism to capture dependencies within items of the same behavior and encode cross-behavior relationships in the input sequence.

Building on these advancements, we propose an attribute- and context-aware multi-behavioral sequential model that effectively leverages multi-behavior data

in a sequential manner. Our model goes beyond traditional methods by integrating item attributes and contextual information, achieving superior performance in unique item recommendation scenarios.

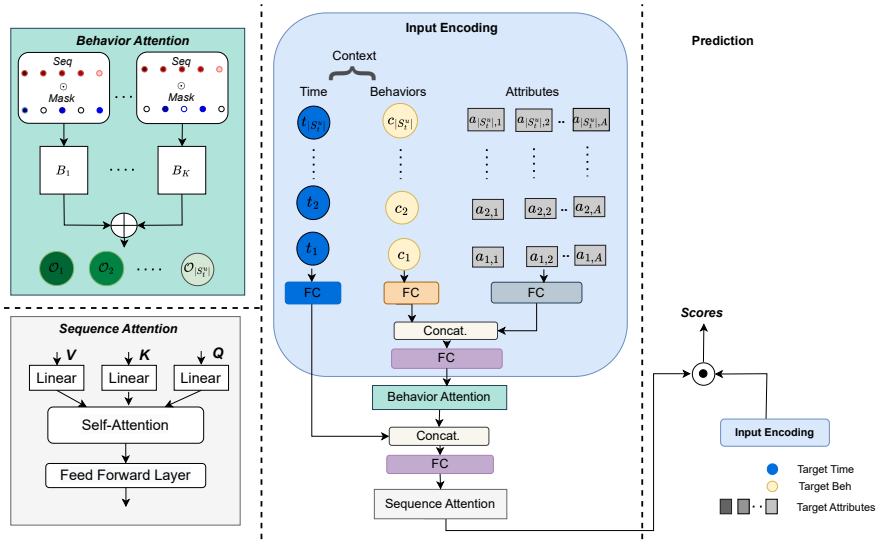

Fig. 1: Attribute and Context-aware Model Architecture

3 Methodology

In this section, we elaborate on the model components, describe the training protocol, model optimization, and provide implementation details.

3.1 Problem Formulation

In the context of unique item sequential recommendation, consider a set of items $\mathcal{I} = 1, \ldots, I$ and a set of users $\mathcal{U} = 1, \ldots, U$. For each user $u \in \mathcal{U}$, there exists a chronologically ordered sequence of item interactions, denoted as $S^u = v_1^u, \ldots, v_{|S^u|}^u$. Each item v is characterized by a set of attributes $\mathcal{A} = 1, \ldots, A$.

Users may engage in up to K distinct behaviors, with each interaction in the sequence corresponding to a specific behavior type. For example, in auction systems, behaviors might include buying, bidding and navigate/click. Each interaction in the sequence is also associated with contextual information, such as the interaction time, denoted as $t_1^u, \ldots, t_{|S^u|}^u$, and the behavior type, represented as $c_1^u, \ldots, c_{|S^u|}^u$.

The primary objective is to predict the next item a user is likely to interact with, focusing specifically on the main behavior. In this case, the buy/sale behavior is considered is the target behavior. Other behaviors act as auxiliary signals to provide a richer representation of user activity over time, enabling more accurate modeling of user preferences.

3.2 Attribute and Context-Aware Multi-Behavior recommendation (ACMB)

Inspired by the multi-behavior recommendation model HMAR [2], we adapt a hierarchical attention mechanism to effectively model diverse user behaviors. Figure 1 illustrates the architecture of the proposed model, which comprises three main components: input encoding, hierarchical masked attention, and prediction for calculating item scores.

Input Encoding In auction systems, items are unique since each is sold only once, making it impractical for the model to rely on item IDs. Instead, each item is represented by its attributes. The input, therefore, consists of two key components; item attributes and interaction context (time and behavior). To derive the latent representation of an item, we first process its attributes vector v_j. The attributes vector of each item in the input sequence are passed through a fully connected layer to generate embeddings as:

$$v'_j = v_j W_v + b_v \tag{1}$$

where $W_v \in \mathbb{R}^{|\mathcal{A}| \times d}$ is the weight matrix, $|\mathcal{A}|$ is the number of items attributes, and d is the items embedding dimension and b_v is the bias term.

A fully connected layer is used for encoding the one-hot vector representing the interaction behavior as follows:

$$c'_j = c_j W_{beh} + b_{beh} \tag{2}$$

where $W_{beh} \in \mathbb{R}^{K \times d_{beh}}$ is the weight matrix, K is the number of behaviors, d_{beh} is the embedding dimension and b_{beh} is the bias term.

Next, items attributes embedding and behavior embeddings are concatenated to form a combined representation q_j. This combined representation is then passed through another fully connected layer to capture a deeper encoding:

$$q_j = \text{concat}_{col}\left(v'_j, c'_j\right), \qquad g_j = q_j W_q + b_q \tag{3}$$

where $W_q \in \mathbb{R}^{(d+d_{beh}) \times d}$ is the weight matrix, d is the layer embedding dimension, and b_q is the bias term. For simplicity, we use the **same dimension** d for all embedding layers.

Attention within behavior Recent approaches often employ graph-based models to capture multi-behavior information. However, this can come at the cost of losing the sequential patterns inherent in the input data. To preserve the chronological order of items while generating latent representations, we draw inspiration from the model proposed in [2] and employ a hierarchical attention mechanism on the input sequence.

The first attention block is designed to encode items associated with a specific behavior. To achieve this, we construct a mask M_b^u from the sequence of items, filtering out items not corresponding to the aimed behavior b. The item sequence embedding $G^u := [g_1, g_2, ..., g_{|S_t^u|}]$ is then multiplied element-wise with the behavior-specific mask $M_b^u := [m_1^b, m_2^b, ..., m_{|S_t^u|}^b]$ as shown in Figure 1.

Ideally, the model employs one behavior encoder block per behavior, resulting in K blocks where K is the number of behaviors. However, when the input sequence includes a large number of behaviors, creating attention blocks for each behavior can lead to excessive memory and computational overhead. To address this, we combine relevant behaviors and process them together through shared behavior attention blocks. The buy-related behaviors are processed through a dedicated attention block, while bid-related behaviors are handled separately in another attention block. Finally, a distinct attention block is assigned to click-related behaviors, ensuring each behavior type is effectively modeled.

Once the input sequence is masked based on behavior type, the masked latent embeddings are passed through a fully connected layer to produce $E_b^u := \{e_1^b, \ldots, e_{|S_t^u|}^b\}$, computed as follows:

$$e_j^b = \left(g_j \odot m_j^b\right) W_e + b_e, \quad W_e \in \mathbb{R}^{d \times d}, \quad b_e \in \mathbb{R}^d \tag{4}$$

Here, $\odot$ denotes element-wise multiplication, W_e is the weight matrix, and b_e is the bias term.

The output embeddings E_b^u, are then fed into a multi-head self-attention block. This block consists of a multi-head attention mechanism followed by a feedforward layer, enabling the sequential encoding of items with the same behavior. The hierarchical attention mechanism ensures that behavior-specific patterns are captured effectively while maintaining the temporal structure of the input sequence.

$$X_b^u = \mathrm{SA}(E_b^u) = \mathrm{concat}_{col}\left(\mathrm{Att}(E_b^u \mathbf{W}_h^{Q^b}, E_b^u \mathbf{W}_h^{K^b}, E_b^u \mathbf{W}_h^{V^b})\right)_{h=1:H} \tag{5}$$

where $\mathbf{W}_h^{Q^b}, \mathbf{W}_h^{K^b}, \mathbf{W}_h^{V^b} \in \mathbb{R}^{d \times \frac{d}{H}}$ represent the linear projection matrices of the head at index h, and H is the number of heads. X_b^u represents the column-wise concatenation of the attention heads.

Finally, we have the point-wise feed-forward layers to obtain the component's final output representations $F_b^u \in \mathbb{R}^{|S_t^u| \times d}$ as follows:

$$\mathrm{F}_b^u = FFN(X_b^u) = \mathrm{concat}_{row}\left(ReLU(X_{b,j}^u \mathbf{W}^{(1)^b} + b^{(1)^b})\mathbf{W}^{(2)^b} + b^{(2)^b}\right)_{j=1:|S_t^u|} \tag{6}$$

where $\mathbf{W}^{(1)^b}$, $\mathbf{W}^{(2)^b} \in \mathbb{R}^{d \times d}$ are the weight matrices of the two feed-forward layers, and $b^{(1)^b}$, $b^{(2)^b} \in \mathbb{R}^d$ are their bias vectors. The initial embedding sequence is added via a ReZero [1] residual connection to the behavior encoder output to obtain as:

$$L_b^u = F_b^u + \gamma \left(G^u \odot M_b^u \right) \tag{7}$$

where γ is a learnable weight initialized with zero to adjust the contribution of $(G^u * M_b^u)$. The output is multiplied by the behavior mask again to mask the positions of the other behaviors as follows:

$$O_b^u = L_b^u \odot M_b^u \tag{8}$$

We combine the output sequences of each behavior $O_b^u := [o_1^b, o_2^b, ..., o_{|S_t^u|}^b]$ by summing them element-wise to generate the first stage multi-behavioral latent sequence embeddings $\mathcal{O}^u$ as follows:

$$\mathcal{O}^u = \sum_{b=0}^{N_{Blocks}} O_b^u \tag{9}$$

where N_{Blocks} is the number of behavior specific attention blocks, O_b^u is the output of the corresponding block.

As previously mentioned, the contextual component of the data consists of two elements: interaction behavior and interaction time. Before applying the behavior-specific attention block, we incorporate the behavior type due to its importance in distinguishing between different behaviors at this stage. To model relationships across the entire sequence and capture interactions between various behaviors, we include time information to indicate the sequential order and provide additional contextual details. Specifically, we extract multiple time-related features from the date, including **year, month, day, week of the year, and day of the year**. These extracted features enrich the sequence representation by providing temporal context.

The time features are encoded using a fully connected layer, producing embeddings that are concatenated with the output of the behavior-specific attention layer. This combined representation is then processed through another fully connected layer, as defined below:

$$t_j' = t_j W_t + b_t \tag{10}$$

$$r_j = LeakyReLu \left(\text{concat}_{col} \left(O_j, t_j' \right) W_r + b_r \right) \tag{11}$$

where $W_t \in \mathbb{R}^{T \times d_t}$ is the weight matrix, T is the number of time extracted details, d_t is the embedding dimension and b_t is the bias term. $W_r \in \mathbb{R}^{(d+d_t) \times d}$ is the weight matrix, d is the layer embedding dimension, and b_r is the bias term. For consistency, we use the same embedding dimension d across all layers. Once the combined representation is obtained, it is passed through the second part of the hierarchical attention mechanism, which models the relationships across different behaviors in the sequence.

Attention across Behaviors Once the sequence comprising all behaviors is constructed, denoted as $\mathcal{R}^u$, the sequence encoder applies attention mechanisms across all items in the sequence. This approach allows the model to capture both intra- and inter-behavioral dependencies, producing a multi-behavior latent representation for each item. The self-attention mechanism is defined as follows:

$$J^u = \text{SA}(\mathcal{R}^u) = \text{concat}_{col}\left(\text{Att}(\mathcal{R}^u\mathbf{W}_h^Q, \mathcal{R}^u\mathbf{W}_h^K, \mathcal{R}^u\mathbf{W}_h^V)\right)_{h=1:H} \tag{12}$$

where $\mathbf{W}_h^Q$, $\mathbf{W}_h^K$, $\mathbf{W}_h^V \in \mathbb{R}^{d\times\frac{d}{H}}$ represent the linear projection matrices of the head at index h, and H is the number of heads. A^u represents the column-wise concatenation of the attention heads. Additionally, for the model stability, we add a residual connection between sequence attention output J^u and sequence items G^u:

$$J^{u\prime} = J^u + G^u \tag{13}$$

Finally, we have the point-wise feed-forward layers to obtain the component's output representations $Z^u \in \mathbb{R}^{|S_t^u|\times d}$ as follows:

$$Z^u = \text{FFN}(J^{u\prime}) = \text{concat}_{row}\left(ReLU(J^{u\prime}\mathbf{W}^{(1)^Z} + b^{(1)^Z})\mathbf{W}^{(2)^Z} + b^{(2)^Z}\right)_{j=1:|S_t^u|} \tag{14}$$

where $\mathbf{W}^{(1)^Z}$, $\mathbf{W}^{(2)^Z} \in \mathbb{R}^{d\times d}$ are the weight matrices of the two feed-forward layers, and $b^{(1)^Z}$, $b^{(2)^Z} \in \mathbb{R}^d$ are their bias vectors.

3.3 Model Prediction and Training protocol

For item ranking, we calculate the final score by taking the dot product of the last item embedding $z_{|S_t^u|}$ from the sequence encoder and the target item embedding $q^o_{|S^u_{t+1}|}$ as follows:

$$\hat{Y}_{t+1} = \sigma(z_{|S_t^u|} \cdot q^o_{|S^u_{t+1}|}) \tag{15}$$

Where σ is a sigmoid function.

In multi-behavior datasets, interactions have varying importance for next-item recommendation. To handle this, we introduce weighting factors α_b in the loss function, one for each behavior. The number and values of α_b may differ based on the dataset's behavior count and their importance. Thus, our weighted binary cross-entropy objective for multi-behavior recommendation is as follows:

$$\mathcal{L}_{rank} = -\sum_{S^u\in\mathcal{S}}\sum_{t=0}^{|S^u|}[\alpha_b log(\hat{Y}_t^{O^{(+)}}) + (log(1-\hat{Y}_t^{O^{(-)}}))] \tag{16}$$

where $\hat{Y}_t^{O^{(+)}}$ are the output scores for the positive samples and $\hat{Y}_t^{O^{(-)}}$ are the output scores for the negative samples, $\mathcal{S}$ is the set of all sequences, α_b indicate the behaviors weights.

Model Training Following the training protocols established by the SASRec [8] and CARCA [11] models, we define a fixed sequence length L for the input sequence. The input sequence is randomly selected [14] from the user's entire item sequence $|S_t^u|$. Depending on the length of the user's sequence, truncation or padding is applied to ensure the sequence length matches L. The target positive item sequence is constructed with the same fixed length L by shifting the input sequence by one position. In contrast, the negative item sequence is generated by randomly sampling L unseen items from the dataset.

3.4 Implementation Details

To deploy the training pipeline, we utilize the AWS Step Functions service for seamless automation. The data on sales, dealers, and auxiliary behaviors is pulled directly from AWS Athena for pre-processing, after which structured interaction tuples and numerical features are formed. Afterwards, the training job is deployed using the previously determined best hyperparameters and the model is then saved on AWS S3. To ensure reliability, we include multiple validation checks at each step of the pipeline. Figure 2 provides an overview of the entire training pipeline.

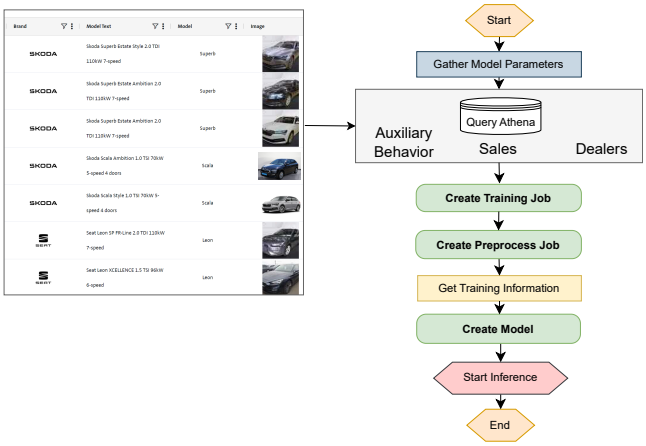

Fig. 2: Model Training Pipeline

Our proposed model is scheduled for deployment in the near future, leveraging AWS SageMaker to precompute all dealer-vehicle scores, which will be stored in Amazon S3 on a weekly basis. When a recommendation request is initiated from the front end for a specific dealer, the API gateway will facilitate the retrieval of the corresponding precomputed scores. Additionally, we intend to conduct a comparative performance analysis between the newly developed ACMB model

and the currently deployed CARCA-based model using A/B testing methodologies. Key performance indicators (KPIs) such as buy-through rate, bid-through rate, and click-through rate will be systematically monitored and evaluated to assess the efficacy of the models.

4 Experiments

In this section, we present the offline evaluation results of our model, including ablation studies to assess the effectiveness of our approach and the importance of individual model components. This analysis aims to address the following research questions:

- RQ1: How does the ACMB model perform compared to attribute-aware sequential recommendation models?
- RQ2: How do auxiliary behaviors impact the model's performance?
- RQ3: How does each model component contribute to overall performance?
- RQ4: How do hyper-parameters influence the model performance?

4.1 Experimental Settings

Dataset The dataset represents a B2B setting where vehicles are sold to dealers. Collected between January 1, 2016, and December 31, 2023, it includes data from 5,916 users (dealers) and documents 1,224,364 sales interactions, each corresponding to a unique vehicle. Each vehicle is characterized by 109 distinct features, such as model, brand, color, and gear type.

In addition to sales data, the dataset contains 4,949,049 bid interactions, which serve as supplementary information for each user. As shown in Table 1, the dataset also includes front-end interaction data collected between December 31 2022 and December 31, 2023 that captures various behaviors within the framework. These behaviors include sale-dialog, which represents a dialog displayed before a sale is finalized, similar to the bid-dialog for bidding interactions. The certificate behavior indicates that the user accessed detailed vehicle information. Furthermore, bookmark behavior, denotes vehicles were saved by users for future reference. Finally, navigate behavior, refers to click interactions on specific vehicles.

All user and dealer data has been anonymized, ensuring confidentiality and compliance with GDPR regulations.

4.2 Evaluation Protocol

We employ a leave-one-out evaluation strategy, where the model is trained and validated using the full interaction sequence, except for the final interaction, which is reserved for testing. To assess performance, we sample 2,000 negative items (N_{test}) from the same month as the corresponding positive item. Model effectiveness is measured using Hit Ratio (HR@N) and Normalized Discounted Cumulative Gain (NDCG@N), with higher values indicating superior performance.

Behavior	Interactions	date
Sale	1,224,362	01.01.2016 -31.12.2023
Bid	4,949,049	
Sale Dialog	171,612	
Bid Dialog	523,191	
Certificate	1,213,960	31.12.2022 -31.12.2023
Bookmark	450,186	
Navigate	4,662,995	
$\sum$ Sum	**13,195,355**	-

Table 1: Dataset statistics

To ensure statistical robustness, we report the mean and standard deviation from three independent runs.

Method	Model type		HR@10	NDCG@10	HR@25	NDCG@25	HR@50	NDCG@50
	Multi-behavior	Sequential						
MultiRec [12]	✓	✗	0.1329 ± $_{2.3e-3}$	0.0917 ± $_{4e-4}$	0.2153± $_{2.7e-3}$	0.0917 ± $_{8e-4}$	0.2941 ± $_{4.3e-3}$	0.1068 ± $_{1.4e-3}$
SASRec [8]	✗	✓	0.1556 ± $_{6e-4}$	0.0740 ± $_{3e-4}$	0.2751 ± $_{1.5e-3}$	0.1031 ± $_{4e-4}$	0.3750 ± $_{1.3e-3}$	0.1223 ± $_{1e-4}$
CARCA [11]	✗	✓	0.1808 ± $_{7e-4}$	0.0966 ± $_{1.1e-3}$	0.2892 ± $_{3.4e-3}$	0.1229 ± $_{4e-4}$	0.3742 ± $_{6.2e-3}$	0.1393 ± $_{3e-4}$
ProxyRCA [14]	✗	✓	0.2135 ± $_{9.2e-3}$	0.1168 ± $_{2.8e-3}$	0.3396 ± $_{8e-4}$	0.1475 ± $_{6e-4}$	0.4477 ± $_{2.9e-3}$	0.1683 ± $_{1e-3}$
ACMB (ours)	✓	✓	0.3304 ± $_{1.5e-3}$	0.1918 ± $_{3e-4}$	0.4742 ± $_{5.7e-3}$	0.2268 ± $_{1.1e-3}$	0.5857 ± $_{3.3e-3}$	0.2483 ± $_{9e-4}$
Improv.(%)			54.75%	64.21%	39.63%	53.76%	30.82%	47.53%

Table 2: Model performance and comparison against baselines on VWFS dataset. The best results are reported in red and the second best in blue.

Baselines We evaluate our proposed method against several attribute-aware sequential recommendation models that rely solely on item attributes, without utilizing item IDs.

- **MultiRec** [12]: A multi-relational model designed specifically for unique item recommendation in auction systems. It leverages item attributes and bid interactions to enhance recommendation accuracy.
- **SASRec** [8]: A self-attention-based model that captures sequential patterns in user interactions. In our setting, we modify the original model by replacing item IDs with item attributes to adapt it for unique item recommendations.

- **CARCA** [11]: A context-aware sequential recommendation model that employs cross-attention between user profiles and item representations for score prediction. For this experiment, we remove item IDs from the model to make it applicable to unique item settings.
- **ProxyRCA** [14]: A state-of-the-art attribute-aware model that introduces proxy-based embeddings, enabling less frequent items to benefit from representations of more frequently occurring ones. In our setup, we exclude item IDs to align with the unique item recommendation scenario.

4.3 Results

Model performance against baselines The unique-item recommendation task is fundamentally challenging, with limited prior research addressing this problem. One notable work is MultiRec [12], a multi-task model that leverages both sales and bid interactions. However, as shown in Table 2, our proposed ACMB model significantly outperforms MultiRec by not only utilizing sales and bid data but also incorporating auxiliary user behaviors through an efficient hierarchical attention mechanism.

Additionally, we compare our model to SASRec [8], a highly effective sequential recommendation model capable of integrating item attributes instead of item IDs. While SASRec focuses solely on the primary behavior, our ACMB model outperforms SASRec, highlighting the importance of incorporating multi-behavioral information to enhance user preference learning.

Furthermore, we evaluate our model against state-of-the-art attribute-aware sequential models, including CARCA [11] and ProxyRCA [14]. While these models effectively leverage item attributes, they lack the ability to incorporate auxiliary behaviors. As demonstrated by our results, ACMB achieves an HR@10 of 0.3304, significantly outperforming CARCA (0.180) and ProxyRCA (0.213). This clearly illustrates the effectiveness and superiority of our proposed model in addressing the attribute- and context-aware multi-behavior recommendation problem.

The effect of auxiliary behaviors on the model performance Auxiliary behaviors provide valuable insights into user preferences; however, their contribution to model performance varies. As shown in Table 3, certain behaviors, such as sale dialog and bid dialog, have a negligible impact, with their effects falling within the standard deviation of the model's results. In contrast, behaviors like bidding and navigation play a more significant role in enhancing performance. This is further supported by their frequency, as bids and navigation interactions are the most dominant behaviors, making them crucial for capturing user intent and improving preference learning.

The effect of model components on the model performance Table 4 highlights the importance of including time information in the input sequence.

Model	HR@10
ACMB w/o Bids	0.3103
ACMB w/o Sale Dialog	0.3273
ACMB w/o Bid Dialog	0.3292
ACMB w/o Certificate	0.3231
ACMB w/o Bookmark	0.3167
ACMB w/o Navigate	0.3127
ACMB w/ Only Sale&Bid	0.2794
ACMB w/ Only Sale	0.2480
ACMB	**0.3304**

Table 3: Auxiliary behaviors effect on the model performance

When time information is excluded, the performance drops to 0.3167, underscoring the critical role of temporal data in capturing interaction patterns. Additionally, the model employs attention blocks to capture dependencies between items within the same behavior. Removing these attention blocks leads to a noticeable decline in performance, with the metric decreasing from 0.3304 to 0.3213. This demonstrates the importance of modeling intra-behavior dependencies for optimal performance.

Finally, we examine the effect of incorporating one-hot encoding of the behavior type as contextual information (BehCxt). While the impact of this component is less pronounced compared to factors like time, it still results in a relative performance drop of 1.8%. This indicates that while behavior type encoding contributes to the model's performance, its influence is relatively minor compared to other components.

Model	HR@10
ACMB w/o Time	0.3167
ACMB w/o BehCxt	0.3243
ACMB w/o BehAtt	0.3213
ACMB	**0.3304**

Table 4: Model components effect on the model performance

The influence of hyper-parameters on the model performance Model hyperparameters play a critical role in determining the performance of a machine learning model. In this work, we investigate the sensitivity of the model to two key hyperparameters: sequence length and embedding size. As illustrated in Figure 3, the sequence length significantly impacts model performance, especially when including several behaviors. Increasing the sequence length allows the model to capture more historical information from the user's behavior, leading to improved performance. However, this comes at a cost: longer sequences

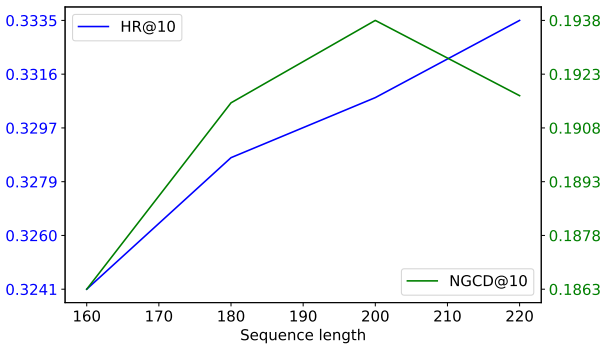

Fig. 3: Effect of sequence length on the HR@10 and NDCG@10

increase the computational complexity and resource requirements of the model. On the other hand, the embedding size is another crucial hyperparameter, as shown in Figure 4. In our dataset, each vehicle is represented by 109 distinct features. A larger embedding size enables the model to better encode these features, enhancing its ability to capture complex patterns. In our experiments, an embedding size of 650 yielded the best results. However, reducing the embedding size to 600 only slightly decreased performance, with the metric dropping from 0.3337 to 0.3275. This suggests that the model is relatively robust to moderate changes in embedding size, allowing for some flexibility in tuning this parameter to balance performance and computational efficiency.

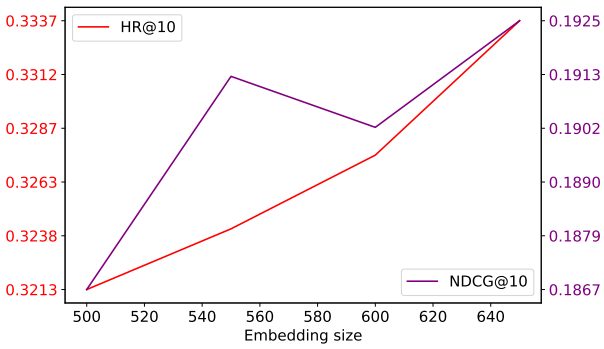

Fig. 4: Effect of embedding size on the HR@10 and NDCG@10

5 Hyperparameters Settings

Our experiments were conducted on an AWS EC2 P3.8xlarge instance equipped with an NVIDIA Tesla V100 GPU, an Intel Xeon E5 CPU, and 244 GB of RAM. We implemented the models using TensorFlow[3]. For hyperparameter optimization, we leveraged AWS Bayesian optimization across the following ranges: latent embedding size [50–700], learning rate [0.000005–0.000025], maximum sequence length [50–250], number of attention heads [1–3], number of blocks [1–3], and dropout rate [0.2–0.6]. The best performance was achieved with a batch size of 128, two attention blocks, and one attention head. The optimal embedding dimension was 667, with a learning rate of 0.0000146, dropout rate of 0.11, and a maximum sequence length of 194. We trained the model for 3,400 epochs. Additionally, we performed a grid search to tune the α_b parameter for different behaviors within the range [0,1]. The optimal values were: 0.8 for sale/buy behavior, 0.5 for sale dialog, 0.2 for bid and bid dialog, and 0.1 for certificate, bookmark, and navigate behaviors. Finally, we used the Adam optimizer [9] for model optimization.

6 Conclusion

In this work, we proposed an attribute- and context-aware multi-behavior (ACMB) recommendation model designed specifically for unique item recommendation in auction systems. Unlike traditional approaches that rely on item IDs, our model leverages item attributes for representation learning. The ACMB model employs a hierarchical attention mechanism, first capturing dependencies among items within the same behavior group, followed by cross-behavior attention to model sequential patterns and auxiliary user behaviors. This approach enhances user preference learning by incorporating richer behavioral context. Experiments on the real-world VWFS dataset demonstrate the superiority of our model compared to state-of-the-art attribute-aware sequential recommendation methods.

References

1. Bachlechner, T., Majumder, B.P., Mao, H., Cottrell, G., McAuley, J.: Rezero is all you need: Fast convergence at large depth. In: Uncertainty in Artificial Intelligence. pp. 1352–1361. PMLR (2021)
2. Elsayed, S., Rashed, A., Schmidt-Thieme, L.: Hmar: Hierarchical masked attention for multi-behaviour recommendation. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining. pp. 131–143 (2024)
3. Elsayed, S., Rashed, A., Schmidt-Thieme, L.: Multi-behavioral sequential recommendation. In: Proceedings of the 18th ACM Conference on Recommender Systems. pp. 902–906 (2024)

[3] https://www.tensorflow.org

4. Guo, H., Tang, R., Ye, Y., Li, Z., He, X.: Deepfm: a factorization-machine based neural network for ctr prediction. arXiv preprint arXiv:1703.04247 (2017)
5. He, X., Chua, T.S.: Neural factorization machines for sparse predictive analytics. In: Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval. pp. 355–364 (2017)
6. Jannach, D., Ludewig, M.: When recurrent neural networks meet the neighborhood for session-based recommendation. In: Proceedings of the eleventh ACM conference on recommender systems. pp. 306–310 (2017)
7. Jin, B., Gao, C., He, X., Jin, D., Li, Y.: Multi-behavior recommendation with graph convolutional networks. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 659–668 (2020)
8. Kang, W.C., McAuley, J.: Self-attentive sequential recommendation. In: 2018 IEEE international conference on data mining (ICDM). pp. 197–206. IEEE (2018)
9. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
10. Li, J., Wang, Y., McAuley, J.: Time interval aware self-attention for sequential recommendation. In: Proceedings of the 13th international conference on web search and data mining. pp. 322–330 (2020)
11. Rashed, A., Elsayed, S., Schmidt-Thieme, L.: Context and attribute-aware sequential recommendation via cross-attention. In: Proceedings of the 16th ACM Conference on Recommender Systems. pp. 71–80 (2022)
12. Rashed, A., Jawed, S., Schmidt-Thieme, L., Hintsches, A.: Multirec: A multi-relational approach for unique item recommendation in auction systems. In: Fourteenth ACM Conference on Recommender Systems. pp. 230–239 (2020)
13. Rendle, S.: Factorization machines. In: 2010 IEEE International conference on data mining. pp. 995–1000. IEEE (2010)
14. Seol, J., Gang, M., Lee, S.g., Park, J.: Proxy-based item representation for attribute and context-aware recommendation. In: Proceedings of the 17th ACM International Conference on Web Search and Data Mining. pp. 616–625 (2024)
15. Sun, F., Liu, J., Wu, J., Pei, C., Lin, X., Ou, W., Jiang, P.: Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In: Proceedings of the 28th ACM international conference on information and knowledge management. pp. 1441–1450 (2019)
16. Tang, J., Wang, K.: Personalized top-n sequential recommendation via convolutional sequence embedding. In: Proceedings of the eleventh ACM international conference on web search and data mining. pp. 565–573 (2018)
17. Wu, L., Li, S., Hsieh, C.J., Sharpnack, J.: Sse-pt: Sequential recommendation via personalized transformer. In: Fourteenth ACM Conference on Recommender Systems. pp. 328–337 (2020)
18. Xia, L., Huang, C., Xu, Y., Dai, P., Zhang, B., Bo, L.: Multiplex behavioral relation learning for recommendation via memory augmented transformer network. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 2397–2406 (2020)
19. Xia, L., Huang, C., Xu, Y., Dai, P., Zhang, X., Yang, H., Pei, J., Bo, L.: Knowledge-enhanced hierarchical graph transformer network for multi-behavior recommendation. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 35, pp. 4486–4493 (2021)
20. Xia, L., Xu, Y., Huang, C., Dai, P., Bo, L.: Graph meta network for multi-behavior recommendation. In: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 757–766 (2021)

21. Yang, Y., Huang, C., Xia, L., Liang, Y., Yu, Y., Li, C.: Multi-behavior hypergraph-enhanced transformer for sequential recommendation. In: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. pp. 2263–2274 (2022)
22. Yuan, E., Guo, W., He, Z., Guo, H., Liu, C., Tang, R.: Multi-behavior sequential transformer recommender. In: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 1642–1652 (2022)
23. Zhou, K., Wang, H., Zhao, W.X., Zhu, Y., Wang, S., Zhang, F., Wang, Z., Wen, J.R.: S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization. In: Proceedings of the 29th ACM International Conference on Information & Knowledge Management. pp. 1893–1902 (2020)